Software work in mathematical biology at Georgia Tech

Maxie Dion Schmidt

maxieds@gmail.com
http://people.math.gatech.edu/~mschmidt34
https://github.com/maxieds

Sandia National Labs
Technical Presentation
Spring 2022

Introduction – Applications of RNA research

- Ribonucleic acid (RNA) sequencing has been utilized in many aspects of cancer research and therapy
- mRNA vaccines have elicited potent immunity against infectious disease targets in animal models of several key viruses (including: flu, Zika, and rabies)
- mRNA vaccines are newly available to the public but have been studied for decades
- This allowed for the rapid development of a COVID-19 virus as soon as data about its RNA from samples was available
- Research on RNA may eventually play a central role in diagnostics, therapeutics and research in medical applications
- As the cellular roles for RNA molecules continue to grow, so does the importance of gaining functional insight from structural analyses

What is mathematical biology?

- Mathematical biology (abbreviated MathBio) is a field that uses mathematical models and theoretical abstractions of the structure of living organisms
- These models are used to explore biological principles dictating the underlying structures, the development and the behavior of these living systems
- In this talk, we will discuss my work as a software engineering RA with the *Georgia Tech Discrete Mathematics and Molecular Biology* (gtDMMB) research group led by Christine Heitsch
- NB: My official title within the group is endearingly designated as the "*Code Goddess*" :)

RNA basics

RNA basics

- RNA is a single-stranded molecule similar to DNA; it carries messenger instructions to the double-stranded DNA that encodes genetic instructions required for life processes
- A strand of RNA has a backbone comprised of alternating sugar (ribose) and phosphate groups
- Each sugar has one of four base types attached to it: Adenine – **A**, uracil – **U**, cytosine – **C**, or guanine – **G**
- Each of the **A**–**U**–**C**–**G** bases can fold to form bonds in pairs

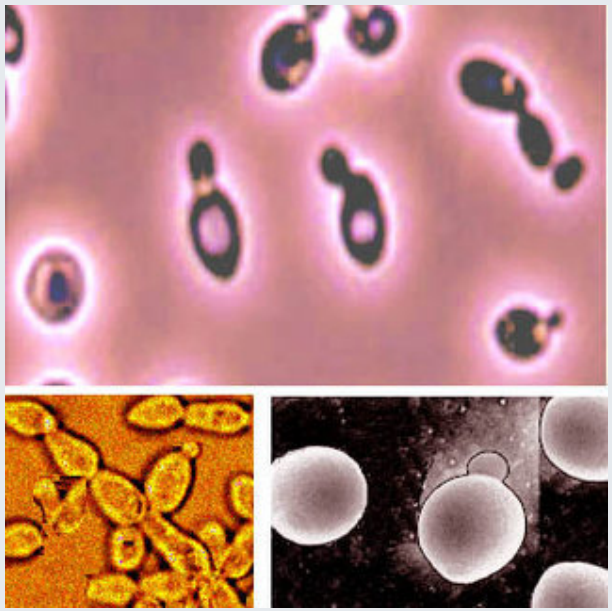
Arc diagrams – Discussion example – S. Cerevisiae (yeast)

GGUUGCGGCCAUAUCUACCAGAAAGCACCGUUUCCCGUCCGAUCAACUGUGUUAAGCUGGUAGA

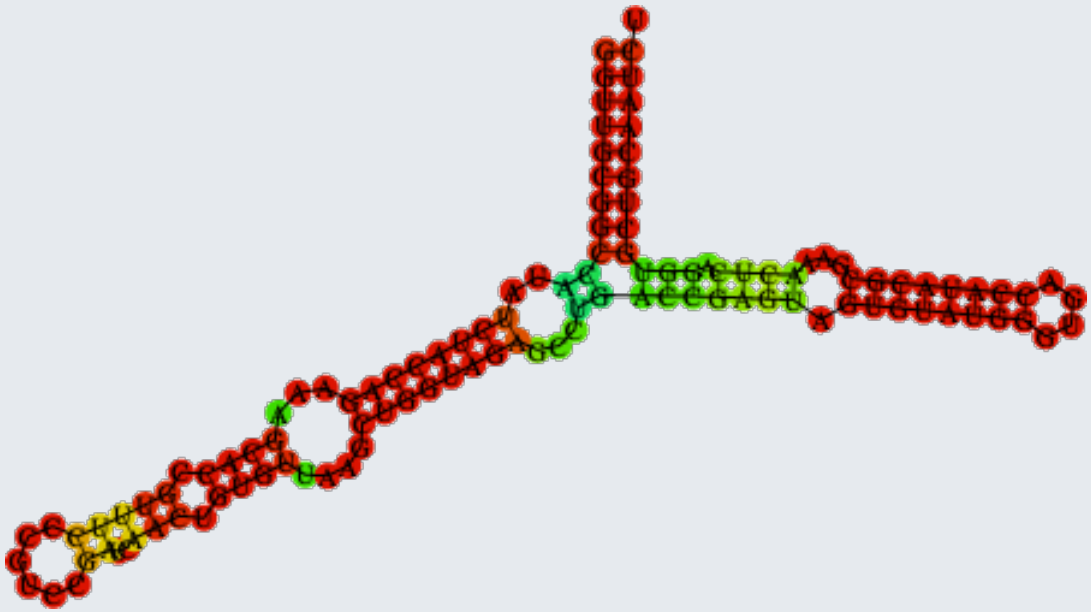
((((((((.....((((((((...((((.((((((.....)).)))).))))....))))))))

GCCUGACCGAGUAGUGUAUGGGUGACCAUACGCGAAACUCAGGUGCUGCAAUCU

.....((((((.(((((((((....)))))))))...))))).)))))))))))).



(Actual microscopic views)



(Radial view of 2D MFE structure)

RNA secondary structures

- A RNA for an organism is defined by the 1D base sequence and can have more than one secondary and tertiary structure by folding
- While obtaining the 1D sequence information is now relatively easy via sequencing, characterizing 3D molecular conformations is still comparatively hard
- Hence, understanding the 2D secondary structures, i.e. the intra-sequence base pairing, remain a crucial component of ribonomics research
- RNA folding prediction programs (e.g., GTFold or RNAFold) can take a base sequence and use probabilistic algorithms to generate the most likely secondary structure
- The secondary structures generated by such computational procedures can be influenced by certain pairing penalties and by environmental constraints like a thermodynamic model specification

GTFoldPython software project

GTFold – Overview I

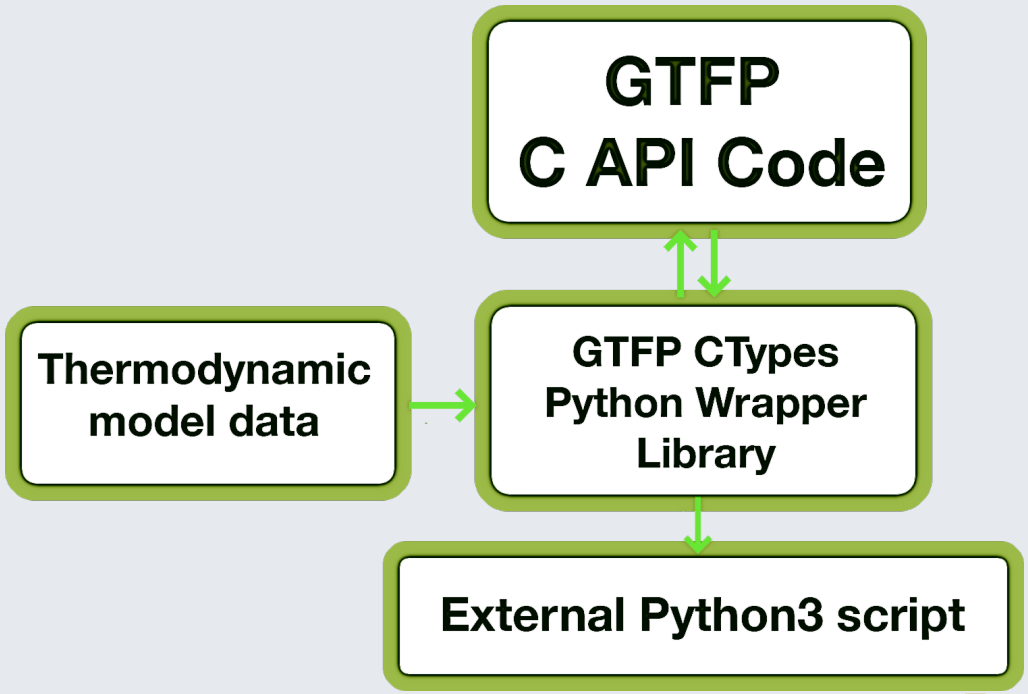
- Accurate and efficient RNA secondary structure prediction remains an important open problem in computational molecular biology
- GTFold is the first implementation of RNA secondary structure prediction by thermodynamic optimization for modern multi-core computers
- **Input:** The base sequence (FASTA) of the organism (optionally: constraints that generated pairings must obey)
  Output: MFE or suboptimal secondary structures (base pair data)
- The difference in the parallel prediction program used by GTFold to take full advantage of today's modern computing technology is particularly valuable to researchers working with very RNA sequences, such as RNA viral genomes

GTFold – Overview II

- The original source code for GTFold produced several command line only utilities that could be used to generate secondary structures (especially, MFE and MFE structures)
- We were motivated by the need to run Python3 on a webserver to index sequences and corresponding computational data generated with GTFold
- One of my ideas as a gtDMMB graduate research assistant in the fall of 2019 was to write Python3 bindings around the original GTFold sources in C++
- Perhaps the most impressive slide of source code in the project is its highly robust and cross platform war-tested `Makefile` and supporting set of install scripts :)

GTFoldPython – Introduction

- The GTFoldPython (GTFP) project provides Python3 bindings based around the original GTFold sources in C++
- The backend uses the Python3 C API
- The frontend is a wrapper library that uses CTypes to call the C API functions from a dynamic shared library (e.g., Windows DLL / MacOS Dylib / Linux SO)



GTFoldPython – Comparison – Python C API code

```c
PyObject * GetMFEStructure(const char *baseSeq, ConsListCType_t consList, int consLength) {
     /* Error checking omitted ... */
     MFEStructRuntimeArgs_t rtArgs;
     InitMFEStructRuntimeArgs(&rtArgs);
     rtArgs.baseSeq = baseSeq;
     SetRTArgsSequenceLength(rtArgs, strlen(baseSeq));
     if(ParseGetMFEStructureArgs(consList, consLength, &rtArgs) != GTFPYTHON_ERRNO_OK) {
          FreeMFEStructRuntimeArgs(&rtArgs);
          return ReturnPythonNone();
     }
     if(InitGTFoldMFEStructureData(&rtArgs) != GTFPYTHON_ERRNO_OK) {
          FreeMFEStructRuntimeArgs(&rtArgs);
          return ReturnPythonNone();
     }
     double mfe = ComputeMFEStructure(&rtArgs);
     if(GetLastErrorCode() != GTFPYTHON_ERRNO_OK) {
          return ReturnPythonNone();
     }
     if(WRITEAUXFILES) {
          ConfigureOutputFileSettings();
          save_ct_file(outputFile, baseSeq, mfe);
     }
     char *dbMFEStruct = ComputeDOTStructureResult(rtArgs.numBases);
     PyObject *mfeTupleRes = PrepareMFETupleResult(mfe, dbMFEStruct);
     Free(dbMFEStruct);
     FreeMFEStructRuntimeArgs(&rtArgs);
     FreeGTFoldMFEStructureData(rtArgs.numBases);
     if(mfeTupleRes == NULL) {
          return ReturnPythonNone();
     }
     return mfeTupleRes;
```

GTFoldPython – Comparison – Wrapper library code

```
## Library initialization code:
if GTFPConfig.PLATFORM_DARWIN:
    GTFoldPython._libGTFoldHandle = ctypes.cdll.LoadLibrary("GTFoldPython.dylib")
else:
    GTFoldPython._libGTFoldHandle = ctypes.PyDLL("GTFoldPython.so",
                                    mode=ctypes.RTLD_GLOBAL, use_errno=True)
@staticmethod
def _WrapCTypesFunction(funcname, restype=None, argtypes=None):
    return GTFoldPython._libGTFoldHandle.__getattr__(funcname)

@staticmethod
def GetMFEStructure(baseSeq, consList = []):
    """Get the MFE and MFE structure (in DOTBracket structure notation)
    :param baseSeq: A string of valid bases (ATGU/X)
    :param consList: A list of constraints on the MFE structure
    :return: A tuple (MFE as double, MFE structure as string in DOTBracket notation)
    :rtype: tuple
    """
    GTFoldPython._ConstructLibGTFold()
    resType = ctypes.py_object
    argTypes = [ GTFPTypes.CStringType,
                 GTFPTypes.FPConstraintsListType(consList),
                 ctypes.c_int ]
    libGTFoldFunc = GTFoldPython._WrapCTypesFunction("GetMFEStructure", resType, argTypes)
    (mfe, mfeStruct) = libGTFoldFunc(GTFPTypes.CString(baseSeq),
                                     GTFPTypes.FPConstraintsList(consList),
                                     len(consList))
    return (float(mfe), str(mfeStruct))
```

GTFoldPython – Example – Find MFE and MFE structure

External Python3 script source:

```
1   import sys, os
2   from GTFoldPythonImportAll import *
3
4   GTFP.Init()
5   GTFP.Config(quiet = False, debugging = False, verbose = False, stdmsgout = "stderr")
6
7   baseSeqFPCons = "GCAUUGGAGAUGGCAUUCCUCCAUUAACAAACCGCUGCGCCCGUAGCAGCUGAUGAUGCCUACAGA"
8   consListFP = GTFPUtils.ReadFPConstraintsFromFile("../Testing/TestData/tRNA/yeast.fa.cons")
9
10  (mfe, mfeDOTStruct) = GTFP.GetMFEStructure(baseSeqFPCons, consListFP)
11  print("MFE␣%1.3f␣=>␣MFE␣DOT␣STRUCT␣\"%s\"\n\n" % (mfe, mfeDOTStruct))
```

Terminal output printed upon invoking the script above:

```
1   MFE -17.200 => MFE DOT STRUCT "(((((((((.........)))).........((((.......)))))...)))))......
```

The RNAStructViz application

RNAStructViz: Graphical base pairing analysis

- The original RNAStructViz application was a project developed by Professor Christine Heitsch and Dr. S. Cheney to visualize RNA secondary structures
- By the time I arrived and started work with gtDMMB in the summer of 2018, the old C++ source code was badly broken with modern Linux and MacOS compilers
- My work was to modernize the C++ source, add support for enhanced graphics using the `cairo` library, and to generally improve and support the project in the long term
- The key feature RNAStructViz provides is visualization and comparisons via arc diagrams of the secondary structures of organisms loaded into the application

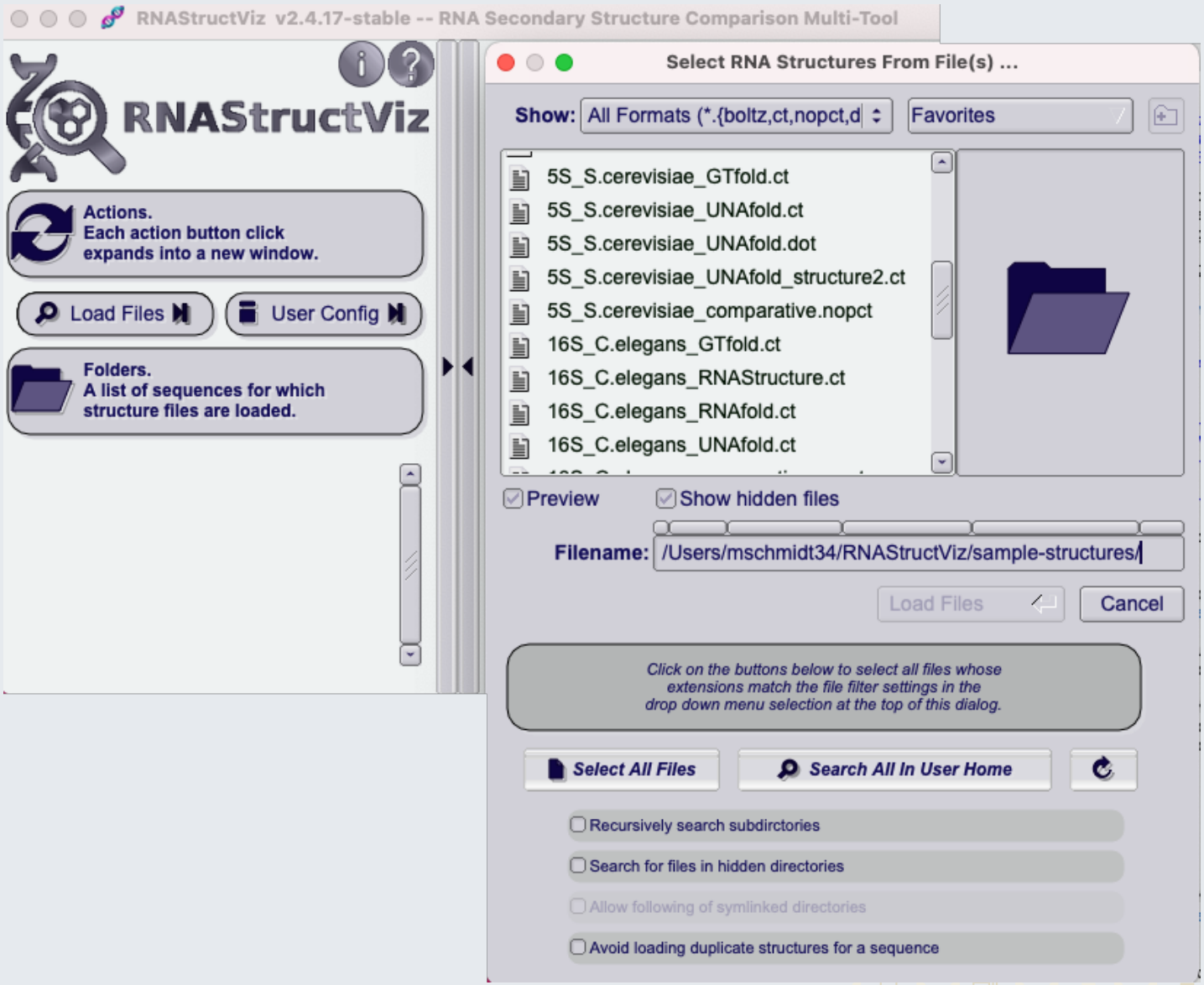
RNAStructViz – Comparison of features

↓ Feature sets	Software support →					
	RNAStructViz	FORNA	jViz.RNA	R-chie	RNAbows	VARNA
❶ — *Platform and availability* —						
Mac OSX support	✔	✓	✓	✓	✓	✓
Linux / Unix support	✔	✓	✓	✓	✓	✓
Windows support	✘	✓	✓	✓	✓	✓
Open source software	✔	✓	✓	✓	✗	✓
Requires external libraries	✔	✓	✓	✓	✗	✓
❷ — *Software usability criteria* —						
Graphical user interface	✔	✓	✓	✓	✓	✓
Web interface	✘	✓	✗	✓	✓	✗
Multi-window interface	✔	✗	✗	✗	✗	✗
Compares 2 structures at once	✔	✓	✗	✓	✓	✗
Compares 3 structures at once	✔	✓*	✗	✗	✗	✗

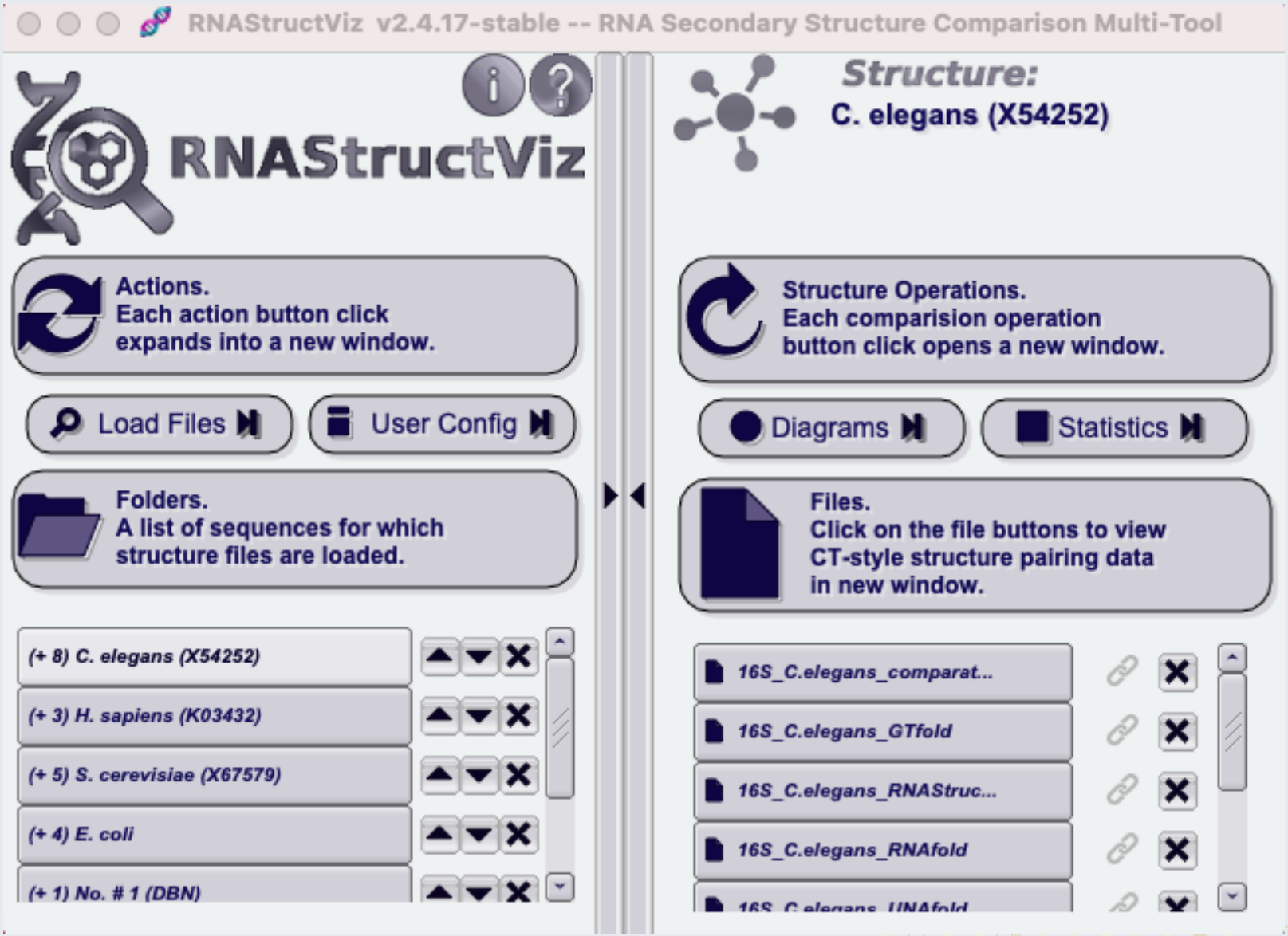
↓ Feature sets	Software support →					
	RNAStructViz	FORNA	jViz.RNA	R-chie	RNAbows	VARNA
❸ — *Support for standard formats* —						
CT files	✔	✗	✓	✓	✗	✓
Dot-bracket files	✔	✗	✓	✓	✗	✓
Built-in file viewer	✔	✗	✗	✗	✗	✓
Requires specialized format	✘	✓	✗	✗	✗	✗
Can edit sequence data	✘	✓	✓	✗	✓	✓
❹ — *Views and diagram type support* —						
Has comparison statistics	✔	✗	✓**	✓	✓	✗
Plots circular arc diagrams	✔	✗	✓	✓	✓	✗
Plots radial diagrams	✔	✓	✓	✗	✗	✓

A comparison of selected features across related tools; an extended survey appears in the RNAStructViz WIKI.

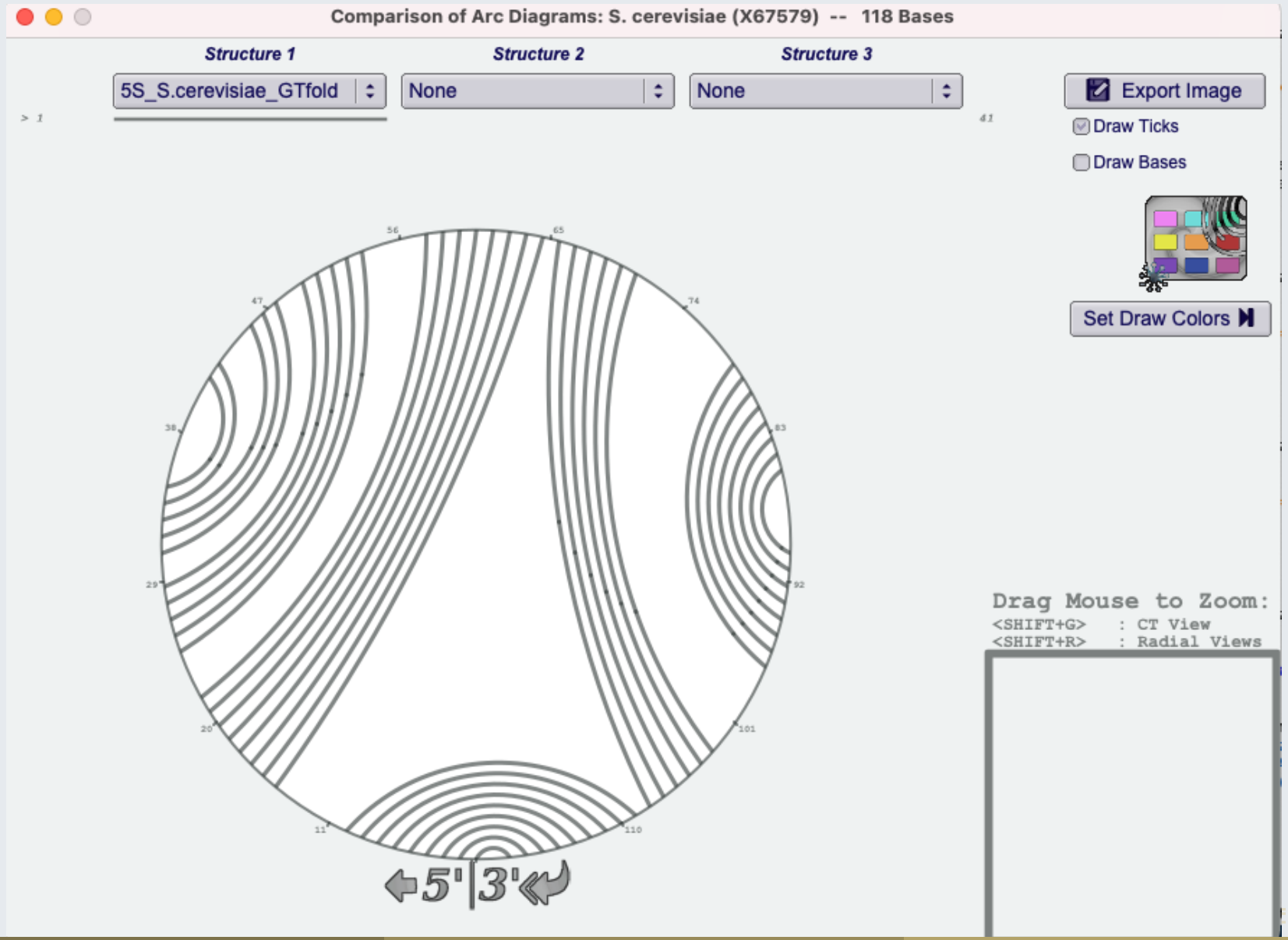
RNAStructViz Screenshot – Loading sample structures I



RNAStructViz Screenshot – Loading sample structures II



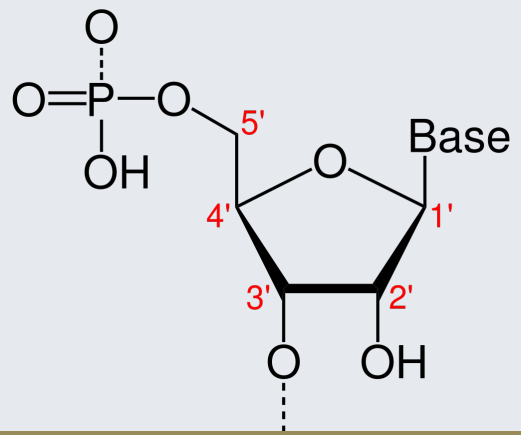
RNAStructViz Screenshot – Arc diagram window



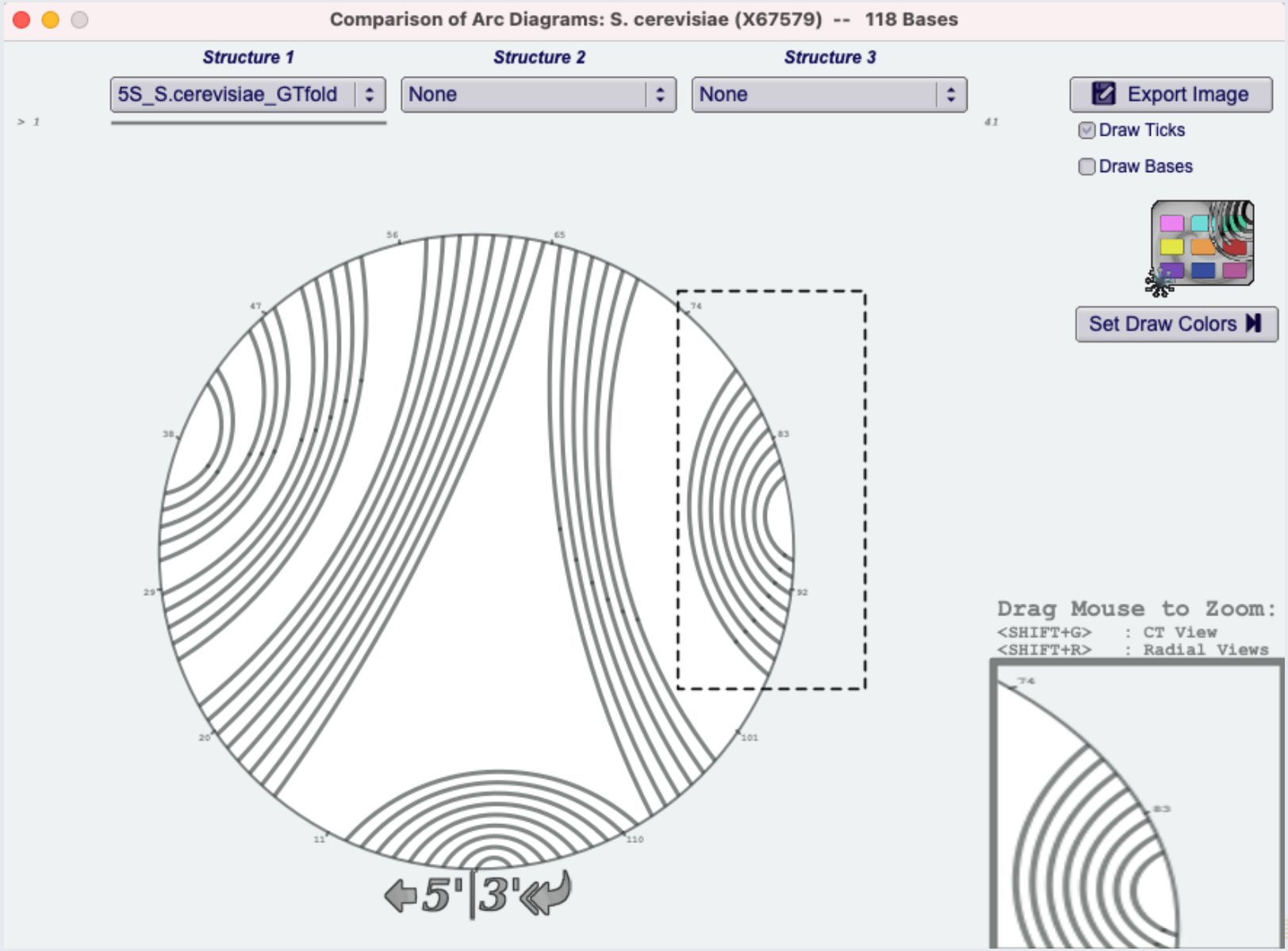
Arc diagram window – Discussion

- The bases indexed from position #1 to #LengthOfBaseSequenceString are placed at equidistant spacings around a circle
- The sequentially numbered base pairs are ordered around the circle counter-clockwise starting from the bottom labeled by the $5' \mid 3'$ directional arrows in the display window
- An arc connecting paired bases is drawn within the circle

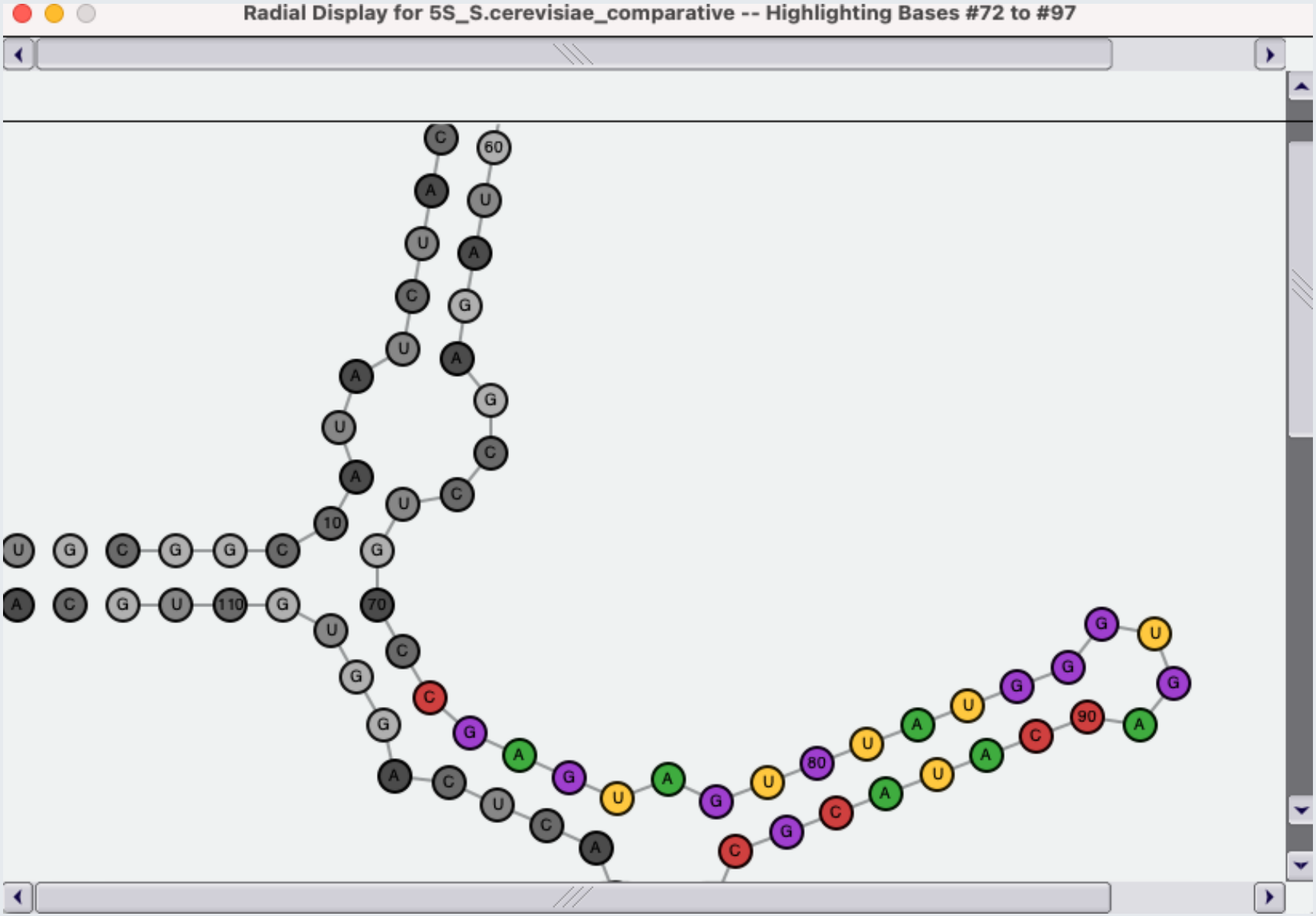
Example: *A furanose (sugar-ring) molecule with carbon atoms labeled using standard notation. The* $5'$ *is upstream, whereas the* $3'$ *is downstream – diagram taken from* WP/Directionality_(molecular_biology)*:*



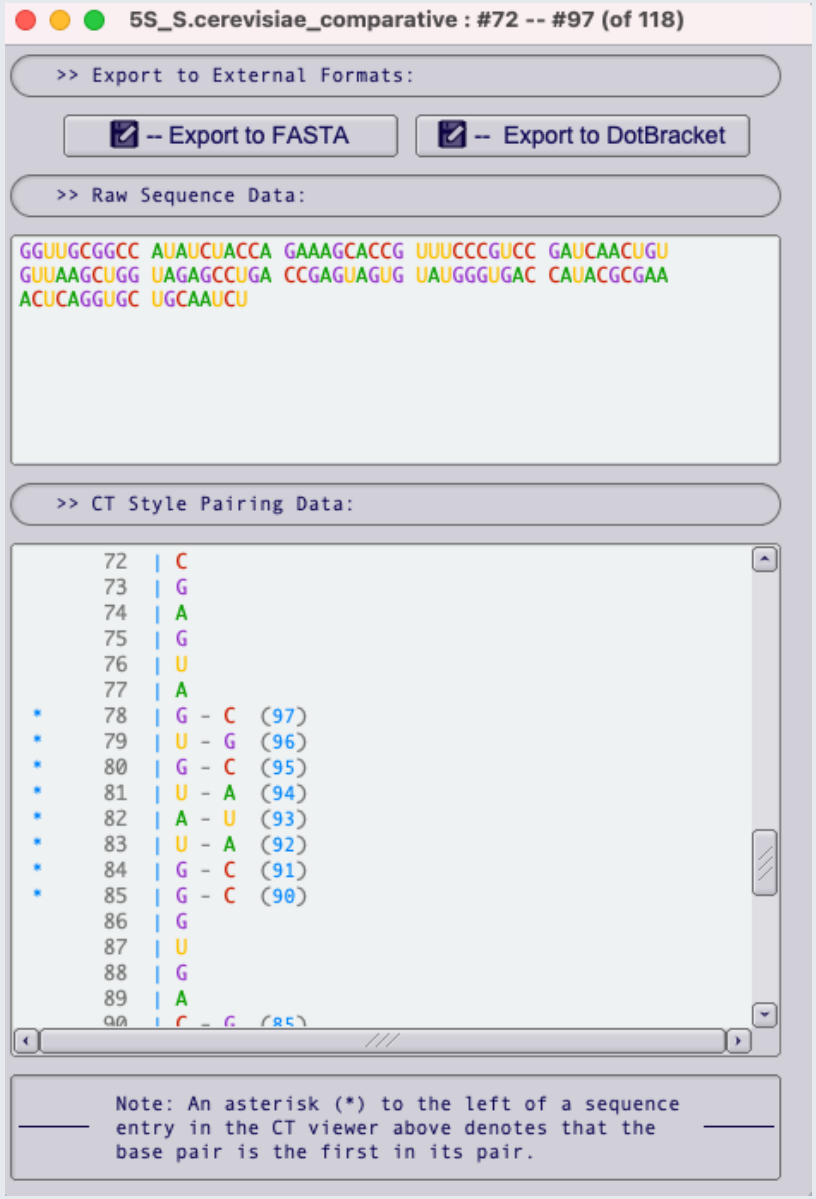
Arc diagram window – Zoom select



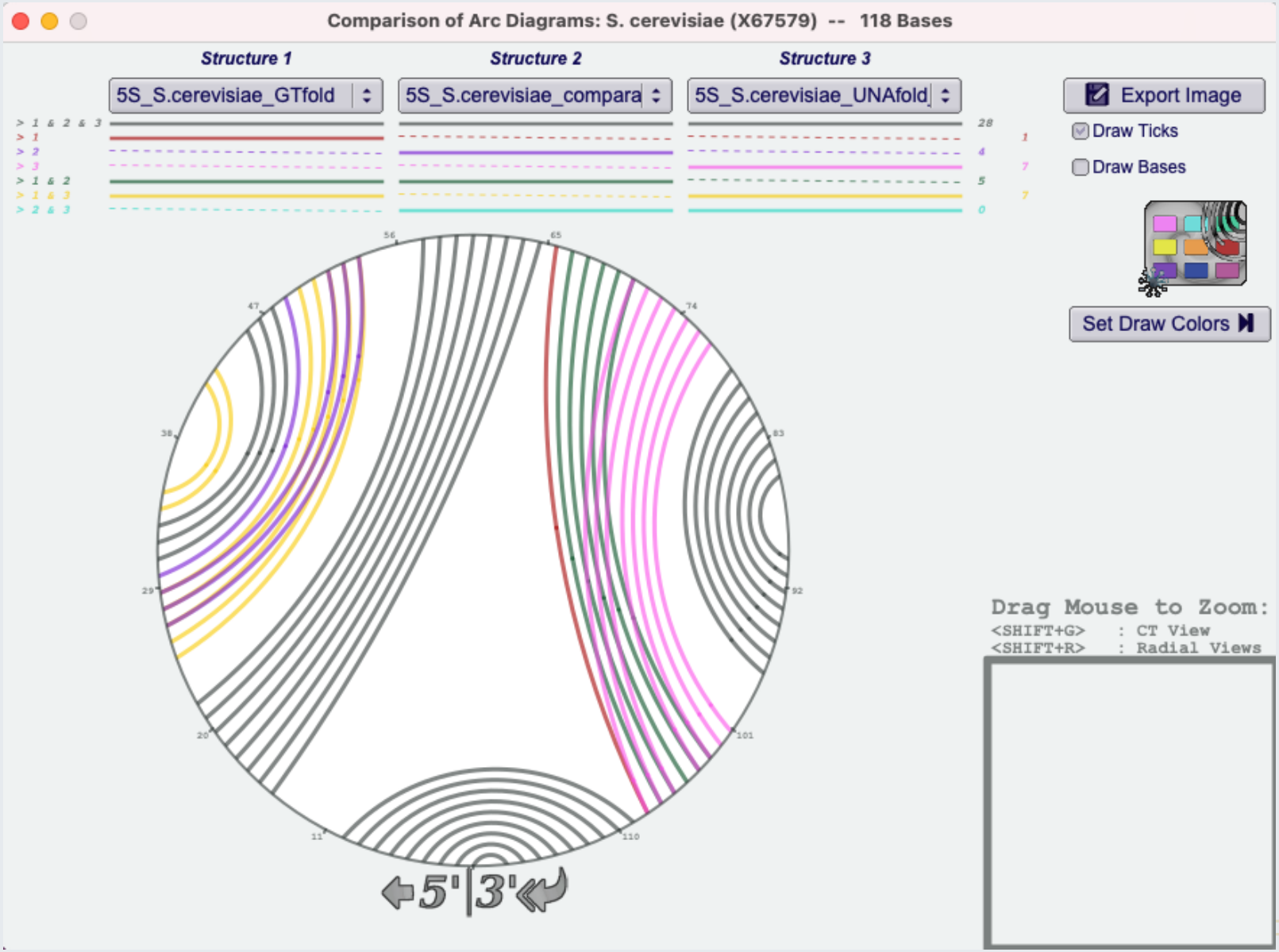
Arc diagram zoom – Radial layout visualization



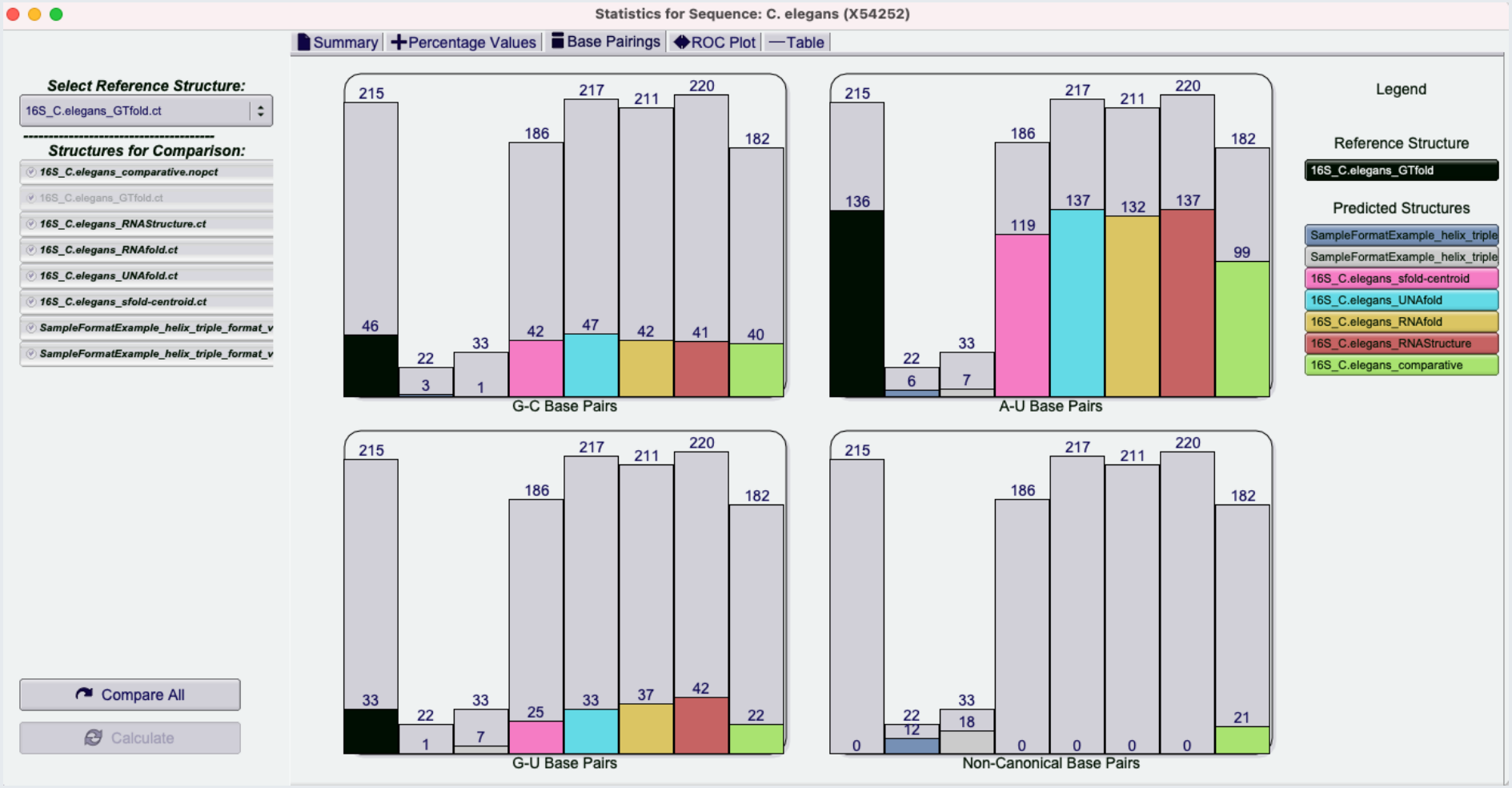
Arc diagram zoom – CT segment visualization



Arc diagram window – Comparing multiple structures



RNAStructViz Screenshot – Statistics window



Summary of accomplishments with gtDMMB software I

- Success in the hundreds of hours spent modernizing and enhancing the source code for gtDMMB projects in computational and mathematical biology
- Success in modernizing and extending build scripts to support installation on MacOS, Linux and Unix-based systems
- A few of the software projects we worked on:
  - RNAStructViz
  - GTFold (CMake builds for MacOS and Linux)
  - GTFoldPython

Summary of accomplishments with gtDMMB software II

- Application note re-introducing our new work on RNAStructViz published in *Bioinformatics* in 2021
- Sister RNA labs that helped with testing and/or use our software include:
  - Computational RNA Genomics Lab at University of California Davis
  - Laederach Lab at the University of North Carolina at Chapel Hill
  - Mathews Lab at the University of Rochester

Concluding remarks

The End

Questions?

Comments?

Feedback?

Thank you for your time!

References I

Centers for Disease Control and Prevention. *Understanding mRNA COVID-19 Vaccines*. Available online: https://www.cdc.gov/coronavirus/2019-ncov/vaccines/different-vaccines/mRNA.html

GTFoldPython software documentation. https://github.com/gtDMMB/GTFoldPython/wiki

RNAStructViz software documentation. https://github.com/gtDMMB/RNAStructViz/wiki

Schmidt, M. D., Kirkpatrick, A., and Heitch, C. *RNAStructViz: graphical base pairing analysis*. Bioinformatics **197** (2021). https://doi.org/10.1101/2021.01.20.427505

Swenson, M. S., Anderson, J., Ash, A., Gaurav, P., Sukos, Z., Bader, D. A., Harvey, S. C., and Heitsch, C. E. *GTfold: Enabling parallel RNA secondary structure prediction on multi-core desktops*. BMC Research Notes. 5(1): 341, 2012. https://github.com/gtDMMB/gtfold